Field Interface Module Software Description

H. Valtier
Software Development Section

The software for the Field Interface Module (FIM) is designed to fully optimize the hardware capabilities of the FIM. Software routines exist which enable the FIM to monitor data, control components, communicate with an uplink computer, and perform real-time and local self-diagnostics. The FIM program is not dependent upon any particular uplink computer and is adaptable to various applications with a minimum of modifications.

I. Introduction

The Field Interface Module (FIM) is a device that functions as a standalone multipurpose controller and data collector. The FIM can control various binary signals and collect (monitor) various digital and analog signals for analysis or interpretation. Furthermore, the FIM can perform real-time self-diagnostics to ensure its own operational integrity. The FIM is designed to interface and communicate with a computer from which it receives real-time commands and to which it can transmit monitored data. The FIM can also perform local self-diagnostics, independent of the uplink computer, by interfacing with a diagnostic tool called the Field Interface Diagnostic Assembly (FIDA).

The topics that will be discussed will provide a description of the architecture and the general capabilities of the FIM from a software standpoint, a brief description of the FIM program, its general I/O characteristics, and some typical applications of the FIM.

II. Architecture

A diagram of the functions and interfaces of the FIM is illustrated in Fig. 1. A system that contains analog sensors (pressure transducers, temperature gages, displacement transducers, etc.) and/or control components (on-off switches, relays, solenoids, etc.) can be monitored and controlled by the FIM. The FIM accepts as its inputs a maximum of 32 digital signals and 32 differential analog signals (three of which are dedicated to internal testing). Furthermore, the FIM will control as its outputs a maximum of 32 digital output lines. A serial I/O communication link is provided for communicating with an uplink computer from which the FIM can receive control commands and to which the FIM can transmit monitored data and its diagnostic status. Finally, the FIM can perform local (as well as real-time) self-diagnostics utilizing the FIDA by temporarily terminating the communication link with the uplink computer.

The FIM is centered around a National Semiconductor BLC 80/204 single board computer (INS8080A-2 CPU). The single board computer contains: (1) 8k of programmable read-only memory (PROM), (2) 4k of random-access memory (RAM), (3) an 8259 programmable interrupt controller (PIC), (4) an H2818 system bus controller (multibus), (5) an 8251 serial I/O interface (USART), (6) two 8255A parallel I/O interfaces, and (7) an 8253 programmable interval timer (Ref. 1). Other boards that comprise the FIM include a Data Translation DT1742 analog-to-digital converter, an Ampex (MCM-8080)

16k expansion memory board, and a multibus-compatible digital I/O board to provide expandable I/O capability.

Local self-diagnostics, independent of the uplink computer, are accomplished by utilizing the FIDA. Basically, the FIDA will allow the FIM to perform the same diagnostics that are performed while the FIM is operating in real-time. However, the FIDA contains a 10-character alphanumeric display for conveying information regarding the status of the PROM, the A/D converter, and the expandable digital I/O board. Local, channel by channel, data monitoring of all digital and analog signals can be performed simply by keying in the desired channel type and number. Routines in the FIM program will enable the FIM to automatically sample the desired channel and dynamically display the data with the proper sign and annotation.

III. Capabilities

The software for the FIM is designed in such a manner as to fully optimize the hardware capabilities of the FIM. The software (approximately 4k bytes) resides in PROM and is automatically activated when power is applied to the FIM. Routines exist which enable the FIM to monitor data, control components, communicate with the uplink computer, and perform real-time and local self-diagnostics.

A. Monitoring Data

The data monitoring portion of the FIM software provides the capability of sampling 16 ac (binary) channels, 16 dc (binary) channels and 32 differential analog channels (12 bits/channel). The software allows the FIM to transmit a full data buffer upon request.

B. Controlling Components

The FIM can be commanded by the uplink computer to control various components by utilizing functions built in to the FIM program to initialize, control, set (ON), and reset (OFF) components. An arming function for the control commands is provided to prevent spurious signals from being issued to the control components. Thus an "arm/set" or "arm/reset" command sequence is necessary to physically turn a control component on or off.

C. Communications

The FIM has the additional capability of communicating with an external computer via the RS232C serial I/O interface. All commands enter the FIM through this interface, which is tied to the interrupt structure of the FIM software. Thus any command that is transmitted by the uplink computer will cause the CPU of the FIM to temporarily suspend all other processing and to recognize, interpret, and process the command. Conversely, all monitored data and diagnostic status information is transmitted by the FIM to the external computer via the RS232C interface. The programmable serial I/O interface is programmed for 7-bit ASCII asynchronous transmission (9600 baud) with single start/stop bits, and odd parity. Normal control commands (initialize, control, set, reset, and arm) will take approximately 60 msec to be interpreted and acted upon. The command to send the entire data buffer will take approximately 245 msec.

D. Diagnostics

Real-time self-diagnostics are performed while the FIM is operating in the REMOTE state and will continuously update its latest diagnostic status. Local self-diagnostics are possible when the FIM is programmed into the LOCAL state and is directly interfaced to the FIDA. The following set of self-tests is performed in either state:

(1) Program (PROM) checksum.

(2) A/D converter channel increment capability.

(3) A/D converter reference voltage (ground, +2.5, and +5.0 volts).

(4) Parallel I/O.

Future enhancements will enable the FIM to perform additional diagnostics regarding:

(5) CPU integrity testing.

(6) RAM bit pattern and latency testing.

IV. Program Description

The overall program flow is diagrammed in Fig. 2, which emphasizes the major control routines. Upon power-up, the FIM initializes the BLC 80/204 single board computer (PWRUPR), then proceeds to initialize the rest of the hardware such as the multibus, interrupt controller, RS232C interface, and parallel I/O interfaces. Finally, the program stack is defined, the RAM is cleared, the operating state of the FIM is set to REMOTE SELF TEST, and the program executes the main program (executive) loop.

In the executive loop (EXEC), the FIM continuously calculates and compares the checksum of the program stored in the PROM. The monitor data (ac, dc, and analog) are also read and updated in a large data buffer in RAM for eventual transmission to the uplink computer. Self-diagnostics are performed and the resulting error status conditions are stored in the large data buffer. The state of the FIM is changed to REMOTE and the next pass through the executive loop is

repeated. Only while the FIM is in the REMOTE state are the interrupts enabled. Thus the uplink computer can communicate with the FIM and transmit commands only in this state. All valid command requests to control, arm/set, and arm/reset any ac and dc control components are executed while the FIM is in the REMOTE state (INTRUP). If the state of the FIM is switched to LOCAL via the FIDA keyboard, the FIDA is directly interfaced to the FIM and local self-diagnostics can be performed. All interrupts are disabled while the FIM is in the LOCAL state and, consequently, commands transmitted by the uplink computer will not be recognized. Switching the state of the FIM to REMOTE via the FIDA keyboard will reestablish communications with the uplink computer.

V. I/O Characteristics

A. Input

All commands that are input to the FIM from the uplink computer must be properly formatted in order to achieve the desired results. The command must commence with a Beginning of Transmission (BOT) character and terminate with an End of Transmission (EOT) character, while the body of the actual command is comprised of 13 bytes which contains information such as the FIM number, request number, request type, command source, and input data. All input commands also contain a command checksum, calculated by the uplink computer, with which the FIM can verify that the command was properly transmitted.

The input request buffer requires 13 bytes to define the body of the request, which is represented in 26 ASCII characters (2 characters/byte). Thus 28 characters (BOT+26+EOT) are required to transmit a proper command to the FIM via the RS232C interface. The desired command is dependent upon the request number, request type, and the input data.

B. Output

Whenever any command is communicated to the FIM from the uplink computer, a response from the FIM is immediately transmitted back (after the requested command is processed) to inform the uplink computer of the outcome of its request. In the case of an invalid command request, the FIM will transmit a Negative Acknowledge (NAK) to signal this condition. For most valid command requests, the appropriate response is governed by a format identical to the input request format with the exception of an additional byte (for a total of 14 bytes) that specifies the current operational state of the FIM. Another exception to the above format concerns the request for the FIM to transmit all of its monitored data and diagnostic status. The response that will be transmitted back is comprised of 103 bytes and will contain the latest diagnostic status information of the FIM (0 = all tests passed, non-0 = last test that failed) as well as the entire set of analog and digital data.

The body of the response message requires either 14 or 103 bytes (as described above), which are represented in 28 or 206 ASCII characters (2 characters/byte). A BOT and an EOT character envelop the response message and thus a total of 30 or 208 characters are transmitted back to the uplink computer to convey the outcome of the input request.

VI. Applications

The FIM program is not dependent upon any particular uplink computer and is adaptable to various applications with a minimum of modifications. Because the FIM functions as a standalone multipurpose controller and data collector, there are potentially several applications for the FIM. Two possible applications will be discussed with regard to (1) The Antenna Control and Monitor (ACM) Subassembly and (2) the Technical Facilities Controller (TFC).

A. ACM/FIM Application

The ACM is planned as a subassembly of the Antenna Control Assembly (ACA), which is in turn a part of the overall MARK IVA DSCC Antenna Mechanical Subsystem (ANT) of the Networks Consolidation Project (NCP). The ACM will provide the capability to monitor and control the safety and integrity of the antenna by the utilization of various sensors and controls dispersed throughout the ANT. The ACM will be configured with a dedicated DSN microcomputer operating in a real-time environment. It will interface with a number of FIMs, via RS232C, to monitor and control the ANT subassemblies. Figure 3 shows the interface block diagram of the ACM and its relationship with the FIM. For the sake of brevity, the other subassemblies that interface with the ACM will not be discussed.

Basically, the FIMs will provide the ACM (uplink computer) with the latest monitor data that is continuously being sampled. Enhancements to the software will enable the FIM to perform comparisons of the monitored data with its applicable operational limits (which will be downloaded by the ACM) and to report any out-of-limit conditions to the ACM. Also, the FIMs will accept commands from the ACM to control certain components on the ANT such as the precharge pump selection switch on the servo hydraulics, the flow switch for the alidade pad recess oil flow rate on the hydrostatic bearing, and the brake set/release limit switch on the gear drive. The monitor data will consist of information such as the oil conditioner outlet temperature, the alidade pad oil film thickness and the accumulator pump inlet pressure of the hydrostatic bearing.

B. TFC/FIM Application

The TFC will perform utility control functions at each of the Goldstone Tracking Stations on a real-time basis based upon programming and upon monitor data accumulated from the sensors distributed throughout the stations. The basic units of the TFC will be small microprocessor units located at each of the tracking stations. These units will collect and perform minimal analysis of data from sensors in the Air Conditioning Group (ACG), Meteorological Monitoring Assembly (MMA), Power Distribution Group (PDG), Civil Structures Group (CSG), Power Generation Group (PGG), Lighting Group (LTHG), Energy Conservation Group (ECG), and the Site Protection Group (SPG). These units will also process data, detect variations from desired conditions, and issue alarms and configuration commands based on monitor data, supplied by the FIMs. A simplified block diagram which illustrates the relationship of the TFC with the FIM is shown in Fig. 4. Again, all other TFC interfaces will not be discussed.

The TFC (uplink computer) provides the capability of scanning all FIMs on a periodic basis for acquiring initial data from the various groups and any subsequent changes to the data. Also, the TFC forwards control commands (received from another processor) to the FIMs and receives messages (monitored data, out-of-limit conditions, or responses to commands) from the FIMs. Typical components that are controlled by the FIMs are such points as the starter battery voltage relay, diesel engine speed relay, and the substation circuit breaker relay of the PGG. Other components controlled by the FIMs include the run status relay of the air conditioning unit, the speed control relay of the air handling unit, and the start/stop control relay of the evaporative cooler of the ACG. The monitor data might consist of information such as the space temperatures of the evaporative coolers, the monitor temperature of the chilled water storage of the ACG, and the fuel level of the main tank of the PGG.

VII. Summary

The Field Interface Module (FIM) is a standalone multipurpose controller and data collector which is designed to interface and communicate with a computer from which it receives commands and to which it can transmit data. The software for the FIM enables it to monitor data, control components, communicate with an uplink computer, and perform real-time and local self-diagnostics. The software for the FIM resides in approximately 4k bytes of PROM which is contained on a BLC 80/204 single board computer. The FIM program provides the capability of sampling 32 digital signals and 32 differential analog signals whose data can be transmitted to an uplink computer upon request. Also, the FIM can be commanded by the uplink computer to control (ON/OFF) up to 32 digital output lines. Communications with an uplink computer is via an RS232C serial I/O interface programmed for asynchronous ASCII transmission at 9600 baud. The FIM software is independent of any particular uplink computer and, with a minimum of modifications, can be utilized in a number of applications.

Acknowledgments

The author wishes to express his appreciation to Tammy Rimmer of the Mini/Microcomputer Applications Group of Section 366 for her initial efforts in developing the FIM Prototype software and related documentation.

Reference

1. Hardware Reference Manual, Pub. No. 420305521-001C, BLC 80/204 Board Level Computer, National Semiconductor Corp., April 1979.

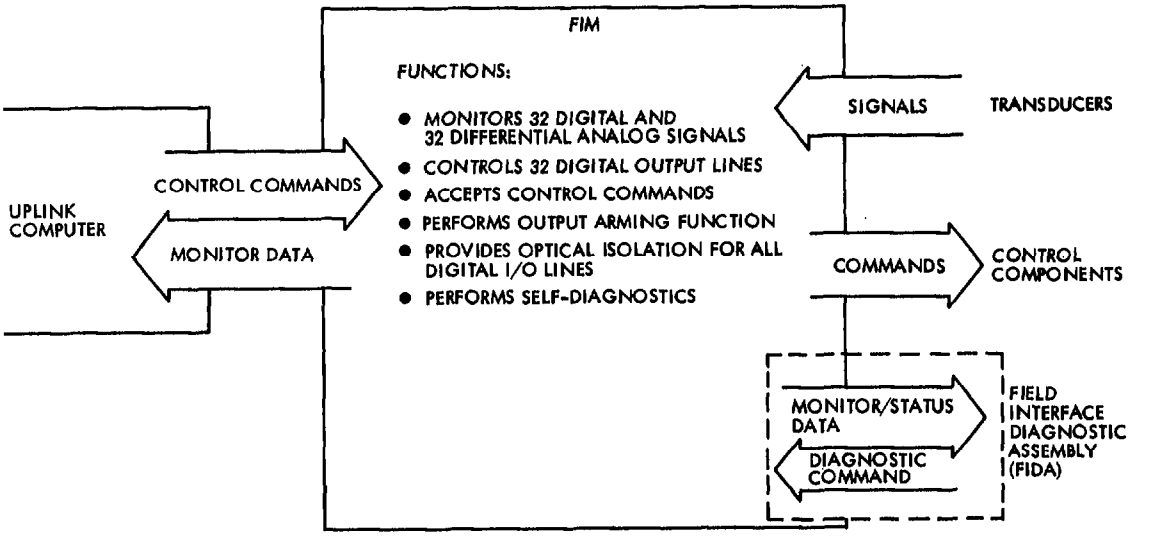

Fig. 1. FIM functions and interfaces

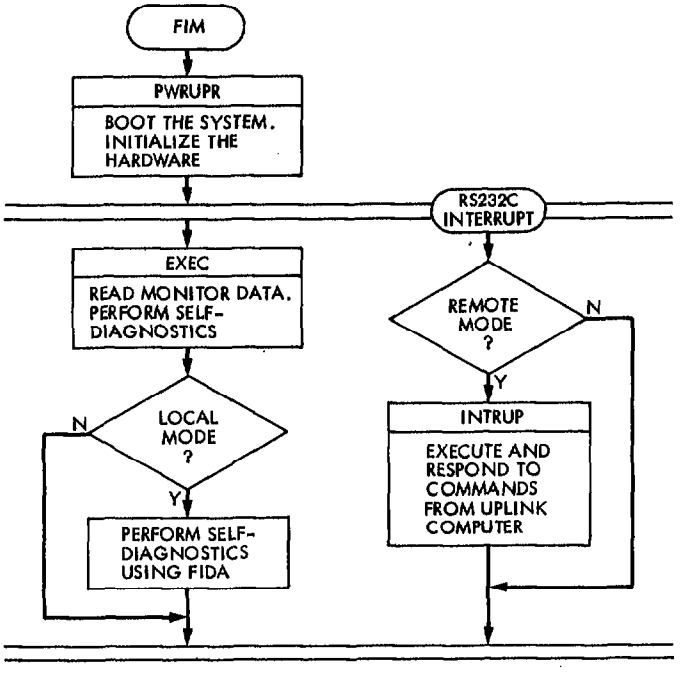

Fig. 2. Control routline flow

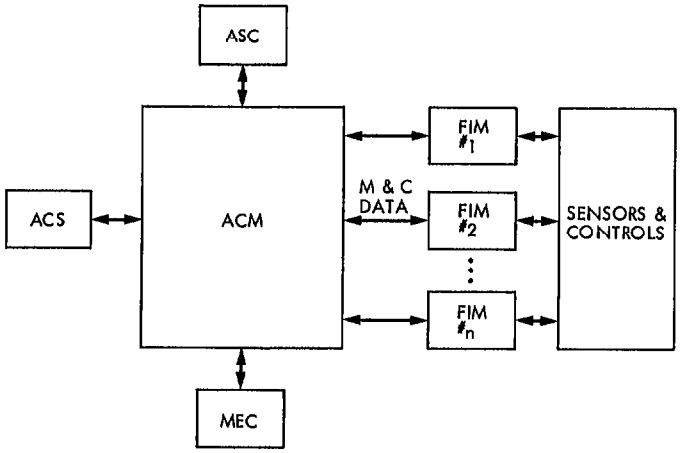

Fig. 3. ACM interface block diagram

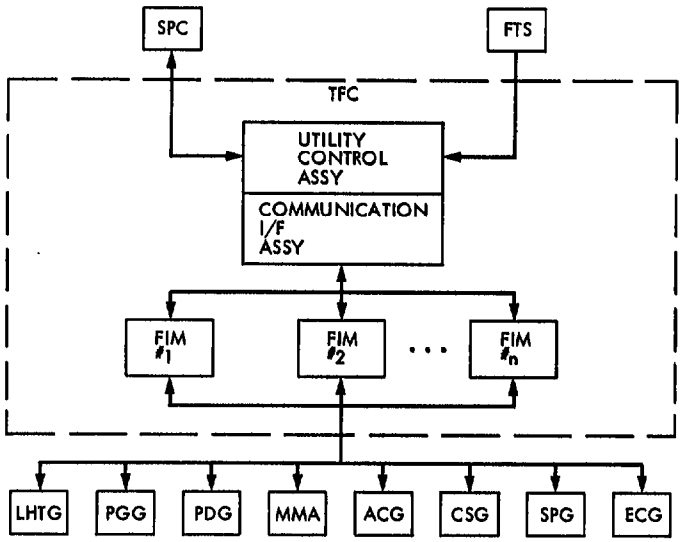

Fig. 4. TFC interface block diagram